WHAT IS CLAIMED IS:

1. A method for extracting a verification model from program source code comprising the steps of:

generating a parse tree defining a control flow from the source code;

identifying source code elements;

from the parse tree, generating source strings for selected ones of the source code elements;

defining corresponding default conversions for translating the source strings into a target language of a model checker; and

generating a verification model in the target language, wherein the verification model conforms to the control flow and to the corresponding default conversions for the selected ones of the source code elements.

2. The method of claim 1 comprising the further steps of:

optionally searching a conversion table for an entry associated with at least one of the source strings, the entry including a translation for the at least one of the source strings; and

substituting the translation for the corresponding default conversion for the at least one of the source strings, wherein the verification model further conforms to the translation.

3. The method of claim 1 wherein the source code elements include basic statements and boolean conditionals.

4. The method of claim 1 wherein the generating of source text strings includes the further step of expressing the source text strings in a canonical form.

5. The method of claim 1 wherein specifics of the corresponding default conversions can depend on a usage of the selected ones of the source code elements.

6. The method of claim 2 wherein the conversion table further includes samples of source strings.

7. The method of claim 2 wherein the conversion table further includes classes of source strings.

8. The method of claim 6 wherein the searching of the conversion table includes the step of pattern matching the at least one of the source strings to the samples of source strings.

9. The method of claim 7 wherein the searching of the conversion table includes the step of pattern matching the at least one of the source strings to the classes of source strings.

10. The method of claim 1 wherein the corresponding default conversions causes the translating of the source strings to respective equivalent statements in the target language when the selected ones of the source code elements are fully relevant to a property to be tested, the translating of the source strings to nul statements in the target language when the selected ones of the source code elements are irrelevant to the property to be tested, and the translating of the source strings to preservation statements in the target language when the selected ones of the source code elements are partially relevant to the property to be tested, preservation statements being statements that preserve a relevant part of the source strings and that suppress an irrelevant part of the source strings.

11. The method of claim 2 wherein the generating a verification model step includes the further step of translating ones of the source strings to a non-deterministic choice of possible outcomes.

12. The method of claim 2 wherein the generating a verification model step includes the step of populating the control flow with the translated source strings.

13. The method of claim 1 wherein the default conversion includes a keep, the keep causing the generating of a verification model step to provide an equivalent statement in the target language.

14. The method of claim 1 wherein the default conversion comprises a hide, the hide causing the generating of a verification model step to provide a nul statement in the target language.

15. The method of claim 1 wherein the default conversion comprises a print, the print causing the generating of a verification model step to embed the respective source strings in a print statement in the target language.

16. The method of claim 2 comprising the further step of simplifying the parse tree according to the translated source strings.

17. The method of claim 16 wherein the simplifying step includes the steps of:

removing nodes corresponding to nul statements;

removing nodes successive to false nodes; and

skipping selected nodess mapped to true.

18. The method of claim 3 comprising the further steps of:

collecting certain data object information for nodes in the parse tree corresponding to basic statements in the source code, the certain data object information including definition information and use information;

constructing a data dependency graph for the source code based upon the collected data object information, the data dependency graph having data dependency graph nodes corresponding to a data object, the data dependency graph having directed edges from first data dependency graph nodes to successive data dependency graph nodes if the successive data dependency graph nodes are used at least once in a definition of the first data dependency graph nodes;

determining a transitive closure for the data dependency graph dependency relation;

adding edges to the data dependency graph according to the transitive closure, the adding step providing a second data dependency graph;

for nodes corresponding to basic statements in the source code having translations other than hide or print, marking second data dependency graph data objects with identifiers corresponding to the definition information and the use information;

for nodes corresponding to basic statements in the source code having a hide translation; marking second data dependency graph data objects with a hide identifier; and

checking the second data dependency graph data objects for identifiers and the hide identifier.

19. A method for verifying that a software based system satisfies certain properties, the software based system having a source code, comprising the steps of:

extracting a finite state model from the source code, the extracting step including the steps of:

abstracting the source code statements based upon relevancies between the certain properties and the source code statements; and

expressing the finite state model in an input language for a model checker; and

checking the finite state model for the certain properties in the model checker.

20. A system for verifying that a system satisfies certain properties, the system having a source code, comprising:

a model extractor operable to extract a finite state model from the source code, the model extractor implementing default conversions for translating selected source code elements and including:

a table of translations for translating other selected source code elements based upon defined abstractions, and

a translator responsive to the translations of the selected source code elements and the other selected source code elements for expressing the finite state model in an input language for a model checker, and

a model checker responsive to the certain properties and the finite state model for checking the finite state model for the certain properties.

21. The system of claim 20 wherein the model extractor further includes a parser for constructing a parse tree from the source code, wherein the translator translates source strings generated from the parse tree.

22. The system of claim 21wherein the model extractor further operates to provide a control flow from the parse tree and to populate the control flow with translated source strings.

23. A method for extracting a verification model from source code having a control flow, comprising the steps of:

generating selected source strings from the source code;

translating ones of the selected source strings to corresponding target language statements according to default conversions;

optionally searching a conversion table for entries associated with the selected source strings, the conversion table including a plurality of translations associated with various ones of the source strings;

translating other ones of the selected source strings to corresponding target language statements according to the entries; and

populating the control flow with the target language statements.